

AMENDMENTS TO THE CLAIMS

Claims 1-6, 8-21, and 23 were originally pending. Please amend claims 16-17, 19-21, and 23. Kindly cancel claim 12. No claims have been added. Accordingly, claims 1-6, 8-11, 13-21, and 23 remain pending. The following listing of claims replaces all prior versions, and listings of claims in the application.

Listing of Claims:

1. (Previously presented) A method to be implemented in a computer system comprising a processor and a memory, the method for managing a run queue comprising a first plurality of threads sorted with respect to one another based on thread priority, the method comprising:

in a deterministic amount of time equivalent to an amount of time to insert a single thread into the run queue, associating a second plurality of threads that is priority sorted with the run queue in a manner that maintains a priority based scheduling semantic of the run queue.

2. (Previously presented) A method as recited in claim 1, wherein the second plurality of threads comprises a root thread, and wherein associating the second plurality of threads with the run queue further comprises inserting only the root thread into the run queue.

1 3. (Previously presented) A method as recited in claim 1, wherein the
2 associating the second plurality of threads with the run queue further comprises
3 inserting each thread in the second plurality of threads into the run queue
4 independent of any additional other queue access.

5
6 4. (Previously presented) A method as recited in claim 1, wherein
7 associating the second plurality of threads with the run queue further
8 comprises inserting only a root thread of the second plurality of threads into the run
9 queue .

10
11 5. (Previously presented) A method as recited in claim 1, wherein
12 associating the second plurality of threads with the run queue further comprises:

13 inserting only a root thread of the second plurality of threads into the run
14 queue; and

15 wherein the method further comprises:

16 removing the root thread from the run queue; and

17 responsive to removing the root thread, inserting a next thread of the
18 second plurality of threads into the run queue such that the priority based
19 scheduling semantic of the run queue is preserved.

1 6. (Previously presented) A method as recited in claim 1, wherein the
2 method further comprises:

3 inserting a root thread of the second plurality of threads into the run queue;
4 removing the root thread from the run queue for execution; and
5 responsive to removing the root thread and independent of any additional
6 other queue access, inserting a next thread of the second plurality of threads into
7 the run queue.

8
9 7. (Canceled).

10
11 8. (Previously presented) A system for managing a run queue, the run
12 queue comprising a first plurality of threads, each thread in the first plurality of
13 threads having a respective priority, the first plurality of threads being sorted such
14 that a thread having a high priority is removed from the run queue before a thread
15 having a lower priority, the system comprising:

16 a memory for storing the run queue and computer-executable instructions;
17 a processor operatively coupled to the memory, the processor being
18 configured to execute the computer-executable instructions for:

19 in an amount of time to insert a single thread into the run queue,
20 associating the second plurality of threads that is priority sorted with the run
21 queue, the associating maintaining a priority based scheduling semantic of the run
22 queue.

1 9. (Previously presented) A system as recited in claim 8, wherein
2 associating the second plurality of threads with the run queue is performed
3 independent of more than a single other queue access.
4

5 10. (Previously presented) A system as recited in claim 8, wherein the
6 second plurality of threads comprises a root thread operatively coupled to one or
7 more other threads of the second plurality of threads, each of the one or more other
8 threads having a respective priority that is a lower priority or an equal priority as
9 compared to a priority of the root thread.
10

11 11. (Previously presented) A system as recited in claim 8, wherein
12 associating the second plurality of threads with the run queue further comprises
13 inserting only a root thread of the second plurality of threads into the run queue.
14

15 12. (Canceled).
16

17 13. (Previously presented) A system as recited in claim 8:
18 wherein the first plurality of threads is a first linked list data structure;
19 wherein the second plurality of threads is a second linked list data structure
20 comprising a root node that is operatively coupled to one or more other threads in
21 the second plurality of threads; and

22 wherein the single insert operation is an operation comprising inserting the
23 root node into a position in the first linked list data structure.
24
25

1 14. (Previously presented) A system as recited in claim 8, wherein
2 associating the second plurality of threads with the run queue further comprises:
3 inserting only a root thread of the second plurality of threads into the run
4 queue; and
5 wherein the method further comprises:
6 removing the root thread from the run queue; and
7 responsive to removing the root thread, inserting a next thread of the
8 second plurality of threads into the run queue such that a priority based scheduling
9 semantic of the run queue is preserved.

10
11 15. (Previously presented) A system as recited in claim 8, wherein the
12 processor is further configured to execute computer program instructions for:
13 inserting a root thread of the second plurality of threads into the run queue;
14 removing the root thread from the run queue for execution; and
15 responsive to removing the root thread and independent of any additional
16 other queue access, inserting a next thread of the second plurality of threads into
17 the run queue.

1 16. (Currently amended) A computer-readable storage medium
2 comprising ~~computer-executable~~ computer-program instructions to manage a run
3 queue of executable threads sorted with respect to one another based on thread
4 priority, the ~~computer-executable~~ computer-program instructions comprising
5 instructions being executable by a processor for:

6 in a deterministic amount of time that is independent of the number of
7 threads in a second plurality of threads that is priority sorted, the deterministic
8 amount of time being a time to insert a single thread into the run queue,
9 associating the second plurality of threads with a first plurality of threads in the
10 run queue in a manner that maintains a priority based scheduling semantic of the
11 run queue.

12
13 17. (Currently amended) A computer-readable storage medium as
14 recited in claim 16, wherein the second plurality of threads comprises a root thread
15 that is operatively coupled to one or more other threads of the second plurality of
16 threads, and wherein the computer-program instructions for associating further
17 comprise instructions for inserting only the root thread into the first plurality of
18 threads.

19
20 18. (Previously presented) A computer-readable storage medium as
21 recited in claim 16, wherein the first plurality of threads is a first linked list data
22 structure, the second plurality of threads is a second linked list data structure
23 comprising a root node that is operatively coupled to one or more other threads in
24 the second plurality of threads, and the deterministic amount of time is a result of a
25 single insert operation to insert the root node into the first linked list data structure.

1
2 19. (Currently amended) A computer-readable storage medium as
3 recited in claim 16, wherein the computer-program instructions for associating
4 further comprise instructions for:

5 inserting only a root thread of the second plurality of threads into the first
6 plurality of threads;

7 and wherein the ~~computer-executable~~ computer-program instructions
8 further comprise instructions for:

9 removing the root thread from the run queue; and

10 responsive to removing the root thread, inserting a next thread of the
11 second plurality of threads into the first plurality of threads in a manner that
12 maintains a priority based scheduling semantic of the run queue .

13
14 20. (Currently amended) A computer-readable storage medium as
15 recited in claim 19, wherein the computer-program instructions for inserting the
16 next thread are performed independent of an other queue.

17
18 21. (Currently amended) A computer-readable storage medium as
19 recited in claim 16, wherein the computer-program instructions for associating
20 further comprise instructions for:

21 inserting a root thread of the second plurality of threads into the first
22 plurality;

23 removing the root thread from the first plurality of threads for execution;

24 and
25

1 responsive to removing the root thread, inserting a next thread of the
2 second plurality of threads into the first plurality of threads independent of any
3 additional access to another different queue.

4
5 22. (Canceled).

6
7 23. (Currently amended) A computer-readable medium comprising
8 computer-program instructions executable by a processor for:

9 managing a run queue with a run queue data structure, the run queue data
10 structure comprising:

11 a first dimension data field comprising a first plurality of threads
12 sorted with respect to thread priority; and

13 a second dimension data field comprising a second plurality of
14 threads sorted based on thread priority, the second plurality of threads comprising
15 a root thread and one or more other threads.

16
17 24. (Canceled).